VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Konzolová monitorovacia utilita pre CISCO switche

Console Monitoring Utility for CISCO switches

2013 Ľubomír Wenzl

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Ľubomír Wenzl**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Konzolová monitorovací utilita pro CISCO switche
Console Monitoring Utility for CISCO Switches

Zásady pro vypracování:

Úkolem práce je navrhnout a realizovat konzolovou aplikaci pro OS Linux využívající knihovnu ncurses, která bude za pomocí protokolu SNMP monitorovat a zobrazovat aktuální informace o přepínači.

1. Seznamte se a stručně popište protokol SNMP.
2. Popište práci s knihovnou ncurses.
3. Navrhněte a realizujte program, který bude fungovat jako konzolová aplikace. Program bude běžet v nekonečné smyčce a bude zobrazovat aktuální informace o stavu switche, jako například vytížení CPU, systémové paměti, datové toky na jednotlivých interface a podobně.
4. Program by měl umožnit zobrazit detailní informace pro konkrétní interface nebo přehled všech interface.
5. Při programování dbejte všech dobrých zásad pro tvorbu programů v OS Linux.
6. Program by měl být dobře konfigurovatelný, stabilní a jednoduchý, tak aby ho bylo možné využívat ve stálém provozu.

Seznam doporučené odborné literatury:

[1] kolektiv autorů. Linux - Dokumentační projekt. 4. vydání. Brno: Computer Press, 2007. ISBN: 978-80-251-1525-1
[2] Geisshirt, Kenneth. Pluggable Authentication Modules: The Definitive Guide to PAM for Linux SysAdmins and C Developers. 1. vydání. Birmingham: Packt Publishing Ltd., 2007. ISBN 978-1-904811-32-9

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Seidl**

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Súhlasím so zverejnením tejto bakalárskej práce podľa požiadavkov čl. 26, odst. 9 *Študijnej a skúšobnej rady pre štúdium v bakalárskych programoch VŠB-TU Ostrava.*

V Bratislave 16. Apríla 2013 .....

Dolu podpísaný Ľubomír Wenzl čestne vyhlasujem, že som bakalársku prácu Konzolová monitorovacia utilita pre CISCO switche vypracoval sám, pod vedením Ing. Dávida Seidla, Ph.D. Prácu som vypracoval na základe poznatkov získaných počas štúdia a použitej literatúry uvedenej ku koncu tejto práce v časti Literatúra.

V Bratislave 16. Apríla 2013 .....

Touto cestou by som rád poďakoval môjmu vedúcemu pánovi Seidlovi za jeho vedenie a odborné rady, ktoré mi pomáhali pri tvorbe tejto práce. Moja vďačnosť patrí aj mojej priateľke a mojím priateľom, ktorí mi s prácou pomohli, pretože bez nich by táto práca nevznikla.

Abstrakt

Práca prináša a systematizuje pohľad do sveta monitoringu, čiže diaľkového sledovania stavov sieťových zariadení a to pomocou SNMP protokolu. Vysvetlím teóriu SNMP protokolu a jeho prípadné použitie v niektorých sieťových nástrojoch. V tejto práci sa ďalej sústreďujem na použité implementačné postupy pri tvorbe SNMP utility sledujúcej stav sieťového prvku, ktorej technológie demonštrujem vytvoreným programom nasadeného v reálnom živote.

Kľúčové slová: monitoring, sieťovanie, SNMP, správa

Abstract

Work brings and systematises a view into the world of the monitoring, or remote monitoring states of network devices using the SNMP protocol. At this work i focus on implementation procedures used in developing SNMP utility watching state of a network element, which technology i demonstrate by creating the program deployed in real life.

Keywords: monitoring, networking, SNMP, management

Zoznam použitých skratiek a symbolov

SNMP	–	simple network managment protokol
TCP	–	transmission control protocol
UDP	–	unreliable datagram protokol
PDU	–	protocol data unit
IOS	–	internetwork operating system
CLI	–	command line interface
OID	–	object identifier
MIB	–	management information base
SMI	–	structure of management information
ASN	–	abstract syntax notation
SLA	–	service level agreement
GUI	–	graphics user interface
MVC	–	model-view-controller

Obsah

1 Úvod **5**

2 Teória protokolu SNMP **6**
2.1 Podstata SNMP protokolu 6
2.2 Komponenty SNMP protokolu 6
2.3 Štruktúra manažmentových informácií 8
2.4 Princíp komunikácie SNMP protokolu 9
2.5 Vývojové verzie protokolu 10

3 Uvedenie SNMP do praxe **12**
3.1 Vyhľadávač manažmentových informácií 12
3.2 Monitorovací nástroj Cacti 13
3.3 Monitorovací nástroj NAGIOS 13
3.4 Monitorovací nástroj PRTG Network Monitor 14
3.5 Hardvérový monitorovací nástroj 14

4 Návrh softvérového modelu **16**
4.1 Špecifikácia požiadaviek 16
4.2 Prípady používania SNMP programu 16
4.3 Sekvenčný diagram pre stiahnutie informácií o portoch 17
4.4 Aktivity diagram 19
4.5 Návrh grafického prostredia 20

5 Implementácia **21**
5.1 Konfigurácia SNMP servera 21
5.2 Implementačný jazyk a použité nástroje 21
5.3 Použité prídavné knižnice 22
5.4 Spôsoby implementácie niektorých funkcií 25
5.5 Prototyp architektúry systému 26
5.6 Kompilácia a spustenie 26
5.7 Perspektíva ďalšieho vývoja 28

6 Záver **29**

7 Literatúra **30**

Prílohy **31**

A Elektronická príloha **32**

Zoznam tabuliek

1 Funkcie SNMP 10
2 Verzie SNMP protokolu 11

Zoznam obrázkov

1 Komponenty SNMP protokolu 7
2 Najvyššia úroveň internetového štandardu MIB 9
3 Diagram USE CASE SNMP nástroja 17
4 Sekvenčný diagram aktualizácií dát 17
5 Aktivity diagram spustenie programu 19
6 Prototyp užívateľského rozhrania 20
7 Obrazovka SNMP programu 27

Zoznam výpisov zdrojového kódu

1 Funkcia na pozastavenie behu vlákna 24
2 Funkcia na vytvorenie SNMP pripojenia 25
3 Funkcia na uloženie informácií o zariadení 25
4 Spustenie SNMP programu s kompiláciou 27

1 Úvod

Pod pojmom monitoring rozumieme neustále sledovanie vlastností dôležitých zariadení pre chod spoločnosti. V prípade výskytu porúch je tak možné skoré reagovanie na vzniknutú udalosť. Ak však spoločnosť nemá tieto monitorovacie funkcie v systéme implementované, dochádza tak k oneskorenému zisteniu dôvodu poruchy a k jej zdĺhavému zisťovaniu, kde nastala. Sieťoví administrátori tak konajú väčšinou intuitívne a pripájajú sa vzdialene na jednotlivé koncové zariadenia a pomocou testovacích krokov zisťujú, kde môže byť problém, postupne od bodu k bodu.

Preto pre ľahšiu správu a diagnostiku vznikol protokol SNMP, ktorý bol implementovaný do sieťových zariadení a umožnil tak jednoduchým spôsobom získať potrebné hodnoty zo zariadenia, ktoré ovplyvňujú jeho funkcionalitu. V práci sa preto budem venovať tomuto protokolu a jeho štruktúre. Aby som pochopil význam fungovania metód poskytovania hodnôt o zariadeniach, je nutné si najprv vysvetliť, z akých častí sa SNMP protokol skladá. Následne si budem môcť popísať, aké vlastne dáta je možné získať a akú majú štruktúru.

Pri analyzovaní SNMP protokolu je pre nás dôležité mať všeobecný prehlad o jeho nasadení a uvedení do praxe. Spoznáme tak, ako sa využíva v reálnom svete a aké sú jeho prínosy do spoločnosti. Vysvetlíme si tak rôzne monitorovacie nástroje, ktoré nám znázorňujú prácu s dátami získaných práve vďaka SNMP protokolu. Nástroje týkajúce sa SNMP protokolu nemusia byť len monitorovacie, ale môžu to byť aj rôzne pomocné programy na prehľadné získanie identifikátora potrebnej vlastnosti, alebo už konkrétnej jej hodnoty.

Vzhľadom na uvedené skutočnosti a fakty budem môcť navrhnúť môj SNMP program, ktorý bude mať niektoré z ich používateľských vlastností. Budem sa tak zaoberať návrhom užívateľského rozhrania a jeho funkčných možností pomocou UML diagramov, kde priblížím užívateľovi, ako by mal program vyzerať a čo by mal dokázať.

Po návrhu softvérového modelu budem mať predstavu, aké ciele sa budem snažiť dosiahnuť a s pomocou akých technológií. Medzi tieto technológie patrí zvolený programovací jazyk a k nemu prídavné knižnice. Každý z týchto nástrojov popíšem a ukážem na vzorových príkladoch ich prípadné použitie v mojom programe. Výsledkom tejto časti práce bude implementácia konzolovej monitorovacej utility, ktorá bude zobrazovať aktuálne informácie o CISCO prepínači, ktoré bude získavať pomocou SNMP protokolu a zobrazovať ich pomocou knižnice NCurses v konzole Linux operačného systému.

2 Teória protokolu SNMP

V tejto kapitole zanalyzujem základné princípy a vlastnosti protokolu SNMP. Popíšem na akej sieťovej vrstve sa prenáša a ako je dôležité poznať, aké typy správ sú posielané pri komunikácií.

Keďže sa jedná o teóriu, tak je dôležité spomenúť, ako sa postupne vylepšoval štandard protokolu, teda, aké boli jeho verzie a akú ma architektúru.

V poslednom rade vysvetlím atribúty, ktoré budem získavať ako hodnoty zo zariadení a kde sa tieto hodnoty uchovávajú.

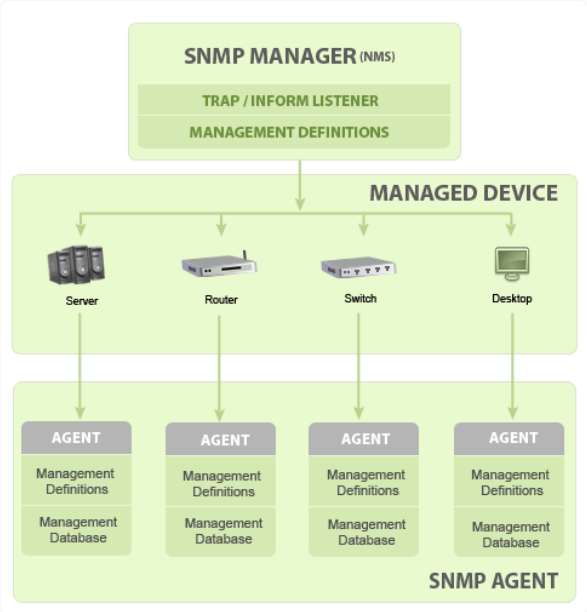
2.1 Podstata SNMP protokolu

Simple Network Management Protocol (SNMP) v preklade znamená jednoduchý protokol pre správu siete. Je to protokol aplikačnej vrstvy, zadefinovaný radou internetovej architektúry podľa štandardu RFC1157 pre výmenu riadiacich a sieťových informácií. Je súčasťou skupiny protokolov TCP/IP.

Tak ako bolo spomenuté už v úvode SNMP je jedným zo široko podporovaných protokolov pre správu a sledovanie sieťových prvkov. Pomocou neho môžu administrátori spravovať a konfigurovať zariadenia v sieti vzdialene z jedného miesta bez použitia softvéru na správu siete. Väčšina z profesionálnej triedy sieťových prvkov musí v sebe obsahovať SNMP agenta. Títo agenti musia byť povolení a nastavení na komunikáciu so sieťovým manažmentom systému pre správu siete (angl. network management system).

2.2 Komponenty SNMP protokolu

V nasledujúcej kapitole vysvetlím ako protokol SNMP, alebo formálne povedané Internet Standard Managment Framework, využíva štruktúru zariadení (angl. managment objects), ktoré sú ovládateľné SNMP agentmi. Títo agenti zobrazujú sieťové operácie SNMP manažérom. Každý SNMP agent obsahuje v sebe databázu nazvanú MIB, ktorá drží v sebe obrovské množstvo dát o operáciách a zariadeniach, ktoré SNMP agent spravuje. Manažér je aplikácia, ktorá tieto dáta zbiera, graficky zobrazuje, alebo riadi. Nasledujúci Obrázok 1 ozrejmuje architektúru a jednotlivé komponenty budú samostatne rozpísané v nasledujúcich podkapitolách.



Obr. 1: Komponenty SNMP protokolu

2.2.1 SNMP manažér

Manažér alebo riadiaci systém je samostatná entita, ktorá je zodpovedná za komunikáciu s SNMP agentmi. Zvyčajne je to počítač, ktorý používa jednu, alebo viac systémov pre sieťový manažment. **Kľúčové funkcie manažéra sú:**

- spracovanie dotazov od agentov
- získavanie odpovedí od agentov
- nastavovanie premenných v agentoch
- akceptuje asynchrónne udalosti z agentov

2.2.2 Manažovacie zariadenia

Manažovacím zariadením sa rozumie sieťový prvok, ktorý je súčasťou siete a vyžaduje určitú formu monitorovania, alebo riadenia. Napríklad smerovače, prepínače, servery, pracovné stanice, tlačiarne, UPS, a iné . . . .

2.2.3 SNMP agent

Agent je program, ktorý je zabalený v sieťovom prvku. Spustenie funkcie agenta umožňuje zhromažďovať databázu riadiacich informácii lokálne na jednom mieste v zariadení a dávať ich k dispozícii SNMP manažérovi, keď si ne o požiada. Títo agenti môžu byť štandardní (napr. Net-SNMP), alebo špecifickí pre výrobcu (napr. HP Insight agent).

Kľúčové funkcie manažéra sú:

- zbiera riadiace informácie o jeho lokálnom prostredí
- ukladá a načítava riadiace informácie zadefinované v databáze manažmentových informácií.
- signalizuje udalosti manažérovi.
- chová sa ako zástupca (angl. proxy) v sieťovom uzle pre niektoré nemanipulovateľné zariadenia, ktoré nepodporujú SNMP.

Zdroj: [2]

2.3 Štruktúra manažmentových informácií

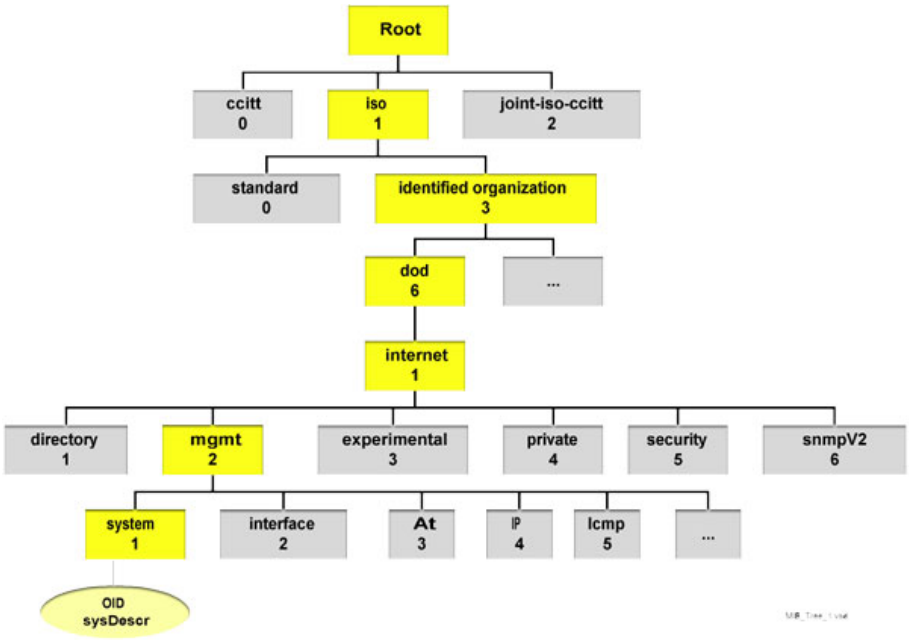
Doteraz som používal termín "manažmentová informácia", ktorá referovala prevádzkové parametre SNMP zariadení. Avšak je nutné si niečo málo povedať z čoho sa riadiaca informácia skladá a čo reprezentuje. Prvý krok k pochopeniu, aký druh informácie zariadenie poskytuje, je pochopiť, v akom kontexte sú dáta reprezentované vo vnútri SNMP. Štruktúra manažmentových informácií nám to presne popíše.

Štruktúra manažmentových informácií (angl. the structure of management information) definuje schému kolekcií spravovaných objektov uložených vo virtuálnom úložisku nazvanom Riadiaca informačná základňa (angl. the management information base), ďalej MIB. Informácie v MIB obsahujú administratívne a prevádzkové informácie o konfigurácii, rovnako ako počítadlá systémových udalostí a činností.

MIB je organizovaná do stromovej hierarchie, v ktorej každý uzol má priradený identifikátor pozostávajúce z kladných celých čísel a voliteľný stručný textový popis. Vrchol MIB sa vzťahuje k riadeniu internetových protokolov a je zhrnutý na Obrázku 2.

Každý manipulovateľný objekt je reprezentovaný koncovým uzlom (listom) a je definovaný svojím menom, syntaxou, režimom prístupu, stavom a popisom. Tak tiež môže byť špecificky identifikovaný jedinečnou pozíciu v strome. Táto pozícia je vyjadrená ako bodkou oddelený rad čiastkových identifikátorov, ktoré začínajú pri koreňovom uzle a končia v pod-identifikátore u konkrétneho objektu koncového uzla. Na Obrázku 2 je uvedený príklad objektu s názvom System, ktorý má jednoznačne identifikovaný reťazec jednotlivých čiastkových identifikátorov 1.3.6.1.2.1.

Zdroj: [1]



Obr. 2: Najvyššia úroveň internetového štandardu MIB

2.4 Princíp komunikácie SNMP protokolu

V predchádzajúcej kapitole som rozoberal, ako SNMP organizuje informácie, ale nateraz musím od toho upustiť a budem sa venovať princípu získavania manažmentových informácií.

2.4.1 Prenos správ SNMP

SNMP je protokol fungujúci na aplikačnej vrstve OSI modelu na komunikačnom princípe klient-server. Namiesto TCP protokolu používa a prenos datagramov „nespoľahlivý“ transportný protokol UDP (definovaný v RFC 768), ktorý je súčasťou skupiny protokolov TCP/IP, má nízke prevádzkové zaťaženie a je rýchly. UDP protokol je jednoduchší a je ho možné naprogramovať i v najjednoduchších zariadeniach. Poskytuje však dostatok funkcií, aby centrálna riadiaca stanica komunikovala so vzdialeným agentom, ktorý je umiestený na ľubovolnom spravovanom zariadení.

Ďalším z dôvodov je nespoľahlivosť, pretože nekontroluje a nečaká na potvrdenie odoslanej správy. Ak nepríde požadovaná odpoveď v určitom časovom cykluse, potom na aplikačnej vrstve agent znova vyšle ďalšiu žiadosť o hodnotu.

Z bezpečnostného hľadiska môže dochádzať k filtrácií TCP správ (angl. packets) na sieti, to by spôsobilo zamedzenie jej manažmentu a jej regulárnej prevádzky. Pri SNMP požiadavkách nieje nutná návratovosť správ. SNMP agent spravidla používa UDP port 161 pre posielanie, prijímanie správ a port 162 na prijímanie trap správ z manažovateľných zariadení.

Zdroj:[3]

2.4.2 Operácie SNMP

SNMPv1 a SNMPv2 protokolové správy (štandard RFC 3419) definujú, ako má manažér a agent, alebo dokonca dva manažéri, odovzdávať informácie. Napríklad, manažér môže použiť tri rôzne správy, na získanie MIB hodnoty z agentov, ktorý pošlú návratovú SNMP Response správu obsahujúcu MIB dáta.

Už z názvu vyplýva, že je jednoduchý a preto má len sedem operácií, alebo správ, ktoré sú uvedené v Tabuľke 1. Tieto správy sú zabalené v štandardizovanom PDU (angl. protocol data unit), ktorý určuje formát posielaných a prijímaných informácií medzi agentom a manažérom.

Stručný popis SNMP operácii je vidieť v nasledujúcej tabuľke.

Správa	Návratová správa	Iniciátor správy	Hlavný účel
Get	Odpoveď	Manažér	Žiadosť na 1 hodnotu
GetNext	Odpoveď	Manažér	Žiadosť na listové hodnoty stromu
GetBulk	Odpoveď	Manažér	Žiadosť na hodnoty podstromu
Response	Nič	Agent	Odpovede na žiadosti
Set	Odpoveď	Manažér	Žiadosť na nastavenie hodnoty
Trap	Nič	Agent	Posiela nevyžiadané správy
Inform	Odpoveď	Manažér	Výmena MIB dát medzi manažérmi

Tabuľka 1: Funkcie SNMP

Zdroj:[4]

2.5 Vývojové verzie protokolu

V tejto časti sa zameriam na rysy rôznych verzií SNMP protokolu a ako bol postupne vyvíjaný štandard.

Vývoj štandardu SNMP bol nedefinovaný na štyri hlavné funkčné oblasti dovoľujúce manažérom riadiť agentov:

- **Definícia Dát** - Syntax ako popisovať dáta agentovi alebo manažérovi. Viac bolo spomenuté v kapitole 2.3.
- **Riadiaca informačná základňa** - Rozširovanie objektov v databáze vzhľadom na proprietárne technológie rôznych výrobcov. Viac bolo spomenuté v kapitole 2.3.
- **Protokoly** - Správy na výmenu dát medzi agentom a manažérom.
- **Bezpečnosť a administratíva** - Definuje zabezpečenie výmeny protokolov.

Zaujímavé je, že týmto oddelením sa dosiahlo, že každá časť bola zlepšená a rozšírená nezávisle v priebehu rokov. Je však dôležité, poznať niektoré z hlavných pridaných prvkov pre každú úradnú SNMP verziu, rovnako, ako aj pre pseudo-verziu zvanú SNMPv2c. Tie najdôležitejšie rozdiely sú popísané v nasledujúcej Tabuľke 2.

Verzia	Popis
1	Používa SMIv1, jednoduchú autentifikáciu cez komunity, ale používa originálnu MIB – I.
2	Používa SMIv2, odobratá potreba komunity, pridaný GetBulk a informačné správy, začiatky originálne MIB – II.
2c	Pseudo vydanie, ktoré povolilo SNMPv1 komunikačný štýl s kombináciou SNMPv2, inak povedané je to ekvivalent SNMPv2.
3	Väčšinou identické s SNMPv2, ale pridaná výrazne lepšia bezpečnosť, hoci zostala spätná kompatibilita s komunitami. Používa MIB - II

Tabuľka 2: Verzie SNMP protokolu

Tabuľka 2 porovnáva rôzne verzie SNMP. Ako sa mohlo očakávať, každá rozširuje predchádzajúcu. Napríklad SNMPv1 definuje reťazec znakov ako názov komunity v podobe čistého textu. SNMPv2 odoberá tuto potrebu komunity, ale necháva spätnú kompatibilitu, ako voliteľný parameter (RFC 1901). Až pri SNMPv3 došlo k zlepšeniu bezpečnosti, ale stále bola spätná kompatibilita cez autentifikáciu pomocou komunity.

Zdroj:[4]

3 Uvedenie SNMP do praxe

Táto kapitola obsahuje prieskum rôzneho použitia protokolu SNMP. Uvádzam, akým spôsobom je možné získať potrebnú hodnotu OID objektu a ako ju jednotlivé aplikačné možnosti využívajú. V predošlej kapitole som vysvetlil, akú má SNMP protokol architektúru, aké ma funkčné vlastnosti a akým spôsobom komunikuje. Uviedol som definície štrukturálnych dát, s ktorými pracuje a taktiež definície operácií, s ktorými ich získava.

3.1 Vyhľadávač manažmentových informácií

Pri vývoji SNMP programov, alebo jednoduchom zisťovaní hodnôt je potrebné poznať OID číslo, alebo presné popisné meno v štruktúre MIB, ktorého je potrebné zistiť hodnotu. Preto nasledujúcej časti uvediem, ako zistiť to správne OID k funkcií čo potrebujem.

Vznikli rôzne programy na uľahčenie orientácie v MIB databáze. Tieto programy majú prehľadný výberový zoznam, kde je možné nalistovať potrebný údaj a zadať požadovanú SNMP funkciu, bez nutnosti poznania OID objektu. Je ich mnoho a uvádzam niekoľko príkladov.

- **SnmpB** - Windows aplikácia zastupujúca rolu SNMP manažéra pod voľnou licenciou. Ide o softvér napísaný v jazyku QT, podporuje všetky SNMP verzie a má možnosť pracovať s MIB databázami. Taktiež podporuje vyhľadávanie SNMP agentov, vie reagovať na trap udalosti a je schopný vygenerovať graf z prijatých údajov.

- **OiDViEW SNMP MIB Browser** - Windows aplikácia umožňuje inžinierom a analytikom ľahko načítať proprietárne MIB súbory, prezerať a manipulovať s dátami, ktoré dáva k dispozícií agent. Dokáže analyzovať SNMP MIB súbory od viacerých dodávateľov, čo neumožňuje žiadna iná aplikácia. Umožňuje prehľadávať štandardizované databáze MIB, bez nutnosti inštalovať aplikáciu z ich web stránky [16].

- **ipMonitor** - Windows aplikácia, ktorá monitoruje sieťové zariadenia. Takisto umožňuje prehľadávať štandardizované MIB databázy, bez nutnosti inštalovať aplikáciu umiestnenej na ich web stránke [17].

- **www stránka výrobcu** - Dosiaľ najväčší výrobca sieťových zariadení spol. CISCO má MIB stromový zoznam OID hodnôt, ktoré podporujú ich zariadenia, umiestený na svojej web stránke [18].

Ďalšou dôležitou funkciou je monitoring (dohľad) počítačovej siete a jej zariadení. SNMP protokol je nasadený v systéme, ktorý funguje neustále a v prípade zaznamenania kritického stavu, poruchy je schopný okamžitej reakcie, napríklad poslaním správy administrátorovi.

Preto sa v nasledujúcich kapitolách zameriam na najpoužívanejšie voľné monitorovacie riešenia, ako sú Cacti, Nagios a platený PRTG network monitor. Väčšina administrátorov sa však uchyľuje k voľným Open-Source systémom. Uvediem aj hardvérovú implementáciu SNMP protokolu do zariadení v podobe čidiel, ktoré snímajú rôzne fyzikálne veličiny.

3.2 Monitorovací nástroj Cacti

Cacti je obľúbený sieťový monitorovací nástroj, založený na grafoch, ktoré využívajú silu ukladania dát pomocou RRDtool a grafického zobrazovania funkčnosti (postavené na PHP). Cacti poskytuje množstvo grafových šablón, metód pre získavanie dát a príjemné užívateľské ovládanie. Toto všetko je zabalené v intuitívnom, ľahko použiteľnom rozhraní, určené pre veľké LAN siete so stovkami zariadení.

Hlavné funkcionality sú združené do štyroch bodov:

1. **Zdroje dát** - dáta sa získavajú pomocou externých skriptov a zhromažďujú sa v MySQL databáze. Ako náhle je zdroj dát vytvorený, dáta sa začnú zbierať po piatich minútach.
2. **Grafy** - po vytvorení zdroja dát je možné ich graficky zobraziť s použitím RRDtool, ktorý predstavuje všetky štandardné typy grafov a funkcií.
3. **Správa užívateľov** - možnosť prideľovania práv užívateľom pre rôzne oblasti grafov, ich nastavení a funkcií.
4. **Šablóny** - vzhľadom k veľkému počtu zdrojov dát a grafov boli vytvorené šablóny, ktoré urýchlili definície akéhokoľvek nového grafu, či dátového zdroja.

Zdroj: [6]

3.3 Monitorovací nástroj NAGIOS

Nagios je výkonný monitorovací systém, ktorý umožňuje organizáciám identifikovať a riešiť problémy v IT infraštruktúre skôr, ako ovplyvní ich kritické obchodné procesy. Je navrhnutý tak, aby bol škálovateľný, flexibilný a prispôsobil sa tak podnikovým procesom.

Hlavné funkcionality sú združené do šiestich bodov:

1. **Monitoring** - monitoruje kritické IT sieťové komponenty, vrátane systémových ukazovateľov, sieťových protokolov, aplikácií, služieb, serverov a sieťovej infraštruktúry.
2. **Varovanie** - posiela upozornenia, keď kritický komponent infraštruktúry zlyhá. Poskytuje tak správcom dôležité oznámenia o udalostiach a to formou SMS, e-mailu alebo vlastného skriptu.
3. **Odpoveď** - IT zamestnanci príjmu oznámenie a okamžite začnú riešiť problém v čom najkratšom čase na dosiahnutie SLA.
4. **Hlásenia** - z historických záznamov je možné vytiahnuť údaje, ako dlho trvali poruchy, udalosti, oznámenia a výstrahy z dôvodu neskoršej kontroly. Na základe týchto údajov je možné overiť, či bolo vaše SLA sú splnené.
5. **Údržba** - počas plánovanej údržby je možné monitoring vypnúť, aby sa neposielali falošné poplachy.
6. **Plánovanie** - Na základe grafov je možné zistiť, či je potrebné vylepšiť zariadenia.

Zdroj: [7]

3.4 Monitorovací nástroj PRTG Network Monitor

Tento profesionálny monitorovací nástroj je však platený, ale zato má obrovské množstvo výhod, z ktorých si zopár uvediem.

- päť rôznych používateľských rozhraní: AJAX web rozhranie, HTML web rozhranie, natívna Windows aplikácia, aplikácia pre iOS a Android,
- podpora viac ako 150 senzorových typov na kontrolu sieťového monitoringu,
- umožňuje nastaviť deväť typov upozornení na udalosti,
- možnosť zdvojenej inštalácie systému, systém sa tvári ako jeden celok a v prípade zlyhania vie automaticky prevziať jeho úlohu druha inštancia,
- prehľadné geografické zobrazenie umiestnenia jednotlivých senzorov, ktoré môžu byť v iných sieťach alebo aj na inom území, všetko je pekne vykreslené na mape,
- export dát do rôznych formátov (HTML, XML, PDF, CSV).

Zdroj: [5]

3.5 Hardvérový monitorovací nástroj

V prechádzajúcich kapitolách som uvádzal softvérovú implementáciu SNMP protokolu, tá však musí vždy odrážať od hardvéru, t.j. v podobe sieťového IP zariadenia. Nemusia to byť len prepínače, servery, smerovače, ale môžu to byť aj senzory (čidlá) rôzneho druhu, napríklad:

- záplavové
- teplotné
- vlhkostné
- spínacie
- požiarne

Tieto čidlá sú poväčšine napojené na základňu, ktorá má už v sebe zabudovaného SNMP agenta s MIB a sieťový port na IP komunikáciu. Ako konkrétny príklad môžem uviesť prípadové štúdie od spoločnosti HW Group.

IT: SOHO aplikácie

- emailové upozornenie na vysokú teplotu (prehriatie).
- emailové, SMS upozornenie pri vlhkosti, teplote mimo povolený rozsah

Serverové miestnosti

- dohľad na dvere rozvádzača (email).

- dohľad na 19‘ rozvádzače po GSM a LAN.
- čítač pulzov pre elektriku, vodu a plyn.

Bezpečnostné aplikácie

- čítač sériovej komunikácie RS-232 pomocou centrálneho IP riešenia.
- spínacie kontakty dverí.

Priemyselné aplikácie

- vzdialené ovládanie relé na 110/230V.
- termostat a hygrostat s web rozhraním.

Zdroj: [8]

4 Návrh softvérového modelu

Po zdokumentovaní teórie SNMP protokolu a jeho uvedení do praxe je vhodné začať s návrhom môjho softvérového produktu, ktorý bude vyžívať SNMP protokol na získavanie dát zo sieťových zariadení. Ďalej v kapitolách zdôraznim, aké sú požiadavky na nástroj a ako sa bude približne so systémom pracovať. V nasledujúcich kapitolách ukážem, za pomoci štandardizovaného modelovacieho jazyka UML, vytvorenie dynamického aj statického diagramu, ktorý popisuje správanie a postup vývoja softvérového modelu.

Pri návrhu softvéru je najdôležitejšie zhrnúť požiadavky na projekt z používateľského a užitočného hľadiska a preto si v nasledujúcej kapitole, ako prvé rozoberiem, špecifikáciu požiadaviek na nástroj.

4.1 Špecifikácia požiadaviek

Účelom vyvíjaného programu bude získavať základné informácie zo sieťových zariadení, konkrétne prepínače od spoločnosti CISCO. Používateľ bude tak môcť sledovať v prehľadnom grafickom prostredí tieto vlastnosti:

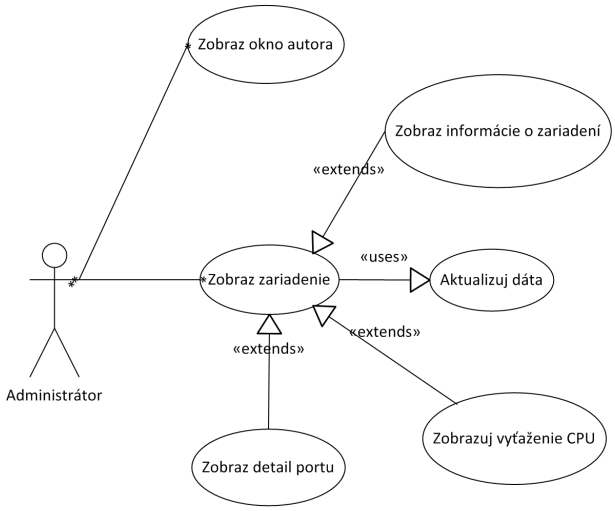
1. využitie procesora.
2. doba zapnutia zariadenia od posledného vypnutia.
3. typ zariadenia.
4. popis zariadenia.
5. podrobné informácie o porte (popis, typ, MAC adresu, pretečené dáta, stav).

Všetky tieto údaje budú mať len informatívny charakter pre sieťového administrátora a nebude ich možné meniť. Avšak umožňujú základnú diagnostiku zariadenia v prípade poruchy sieťovej infraštruktúry, bez nutnosti používania príkazov.

Jednotlivé používateľské možnosti ukážem v nasledujúcej podkapitole za pomoci Use Case diagramu.

4.2 Prípady používania SNMP programu

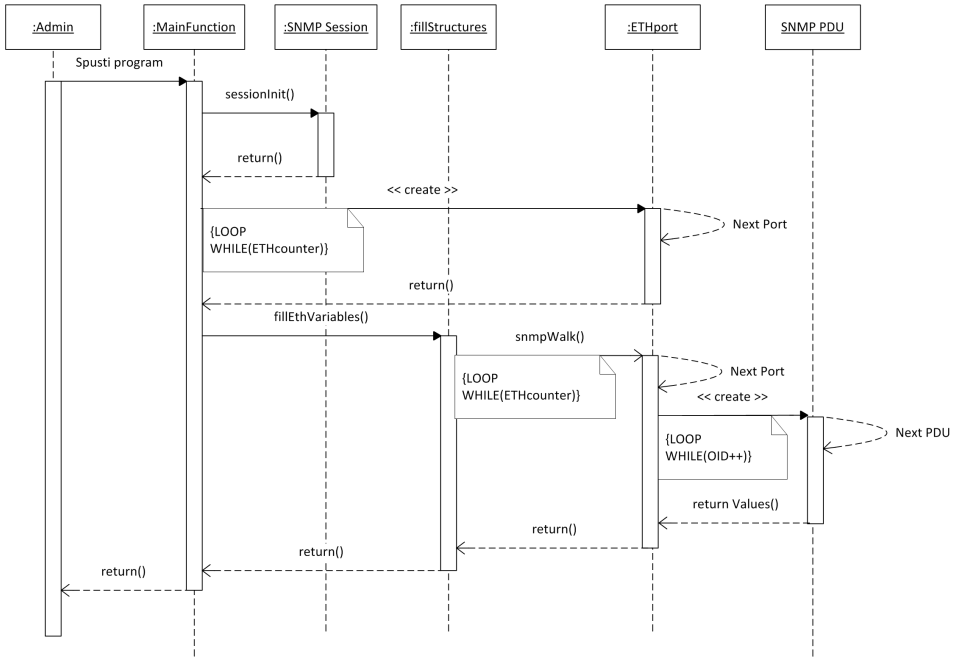
Na Obrázku 3 uvádzam vymodelovaný diagram Use Case. Z obrázku je vidno, že program bude používať len používateľ v jednej roly a to konkrétne sieťový administrátor a nebude nutné definovať prístupové oprávnenia k iným rolám, ktoré by mali obmedzený prístup k jednotlivým funkciám. Tento používateľ potrebuje získať informácie o zariadení, ale nemusí si ich vždy zobraziť všetky (tie sú znázornené pomocou šípky extends). Avšak pri spustení programu je nevyhnutné, aby sa vždy aktualizovali dáta a graficky zobrazil kontrolované zariadenie. Pre tieto prípady sa používa znázornenie pomocou šípky uses.



Obr. 3: Diagram USE CASE SNMP nástroja

4.3 Sekvenčný diagram pre stiahnutie informácií o portoch

Tu uvádzam pohľad na dynamické správanie programu pri štarte, pri ktorom dochádza k volaniu funkcii za účelom aktualizácií dát. Obrázok 4 ilustruje sekvenčný diagram, ktorý uvádza dynamický pohľad na volanie funkcií z časového hľadiska, na proces programu pri dotazovaní o SNMP hodnoty zo zariadenia.



Obr. 4: Sekvenčný diagram aktualizácií dát

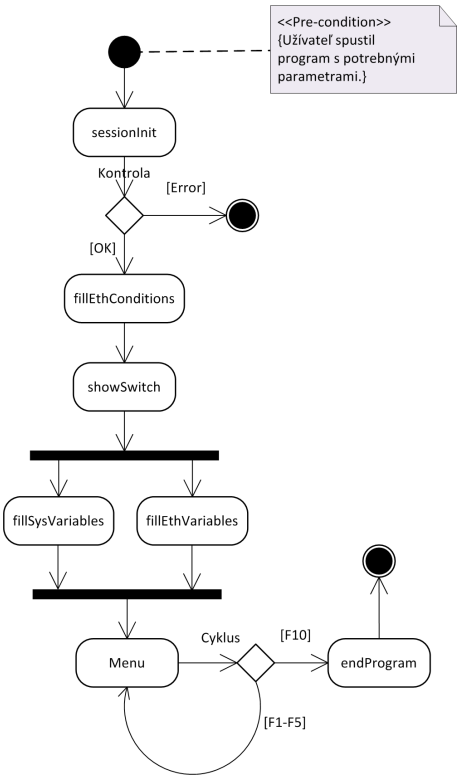
Diagram modeluje situáciu, keď administrátor spustí program, čiže hlavnú funkciu a tá následne vytvorí objekt SNMP session. Na ukladanie hodnôt z SNMP PDU správ je najprv dôležité vytvoriť objekty ETHport. Tento krok sa v sekvenčnom diagrame značí plnou šípkou s príznakom « create » smerujúcu na vytváraný objekt. Tento proces mi volá hlavná funkcia a keďže mám niekoľko portov, musí sa to diať v cykle, ktorý sa opakuje, pokiaľ sa nevytvoria všetky porty. Tento jav sa značí spätnou prerušovanou šípkou.

Po vytvorení poľa štruktúr portov je možné ich naplniť hodnotami. Preto ďalej hlavný proces volá plniacu funkciu `fillStructures()`, ktorá v rámci svojho behu, volá v opakujúcom cykle funkciu `snmpWalk()`, pre každú chcenú vlastnosť portov. Táto funkcia zisťuje vždy jednu konkrétnu vlastnosť skupiny portov, kde na každý port je vytvorené jedno samostatné PDU, v ktorom sa prenášajú manažmentové informácie (opísané v kapitole 2.3). Taktiež sa to deje v opakovanom cykluse, v rámci plniacej funkcie `fillStructures()`, pokiaľ sa neprejdu všetky porty. Tento cyklus imituje SNMP operáciu GetNext podľa Tabuľky 1 opísanej v kapitole 2.4.2.

Návratové hodnoty jednotlivých funkcií sa značia prerušovanou šípkou, kde sa predávajú volajúcim funkciám.

4.4 Aktivity diagram

Obrázok 5 nám ilustruje aktivity diagram znázorňujúci spustenie programu z časového hľadiska. Podobne ako sekvenčný diagram aj aktivity diagram vyjadruje dynamický pohľad behu programu.



Obr. 5: Aktivity diagram spustenie programu

V tomto diagrame som sa zameral na hlavné funkčné kroky, ktorými program prechádza v jednotlivých časových úsekoch. Každý krok (aktivita) je v zaoblenom rámčeku a nadväzuje na predchádzajúci. Spustenie programu je podmienené zadaním správnych parametrov a to sa v aktivity diagramoch znázorňuje v obdĺžniku s nadpisom «Pre-conditions» a vypísaním potrebnej podmienky.

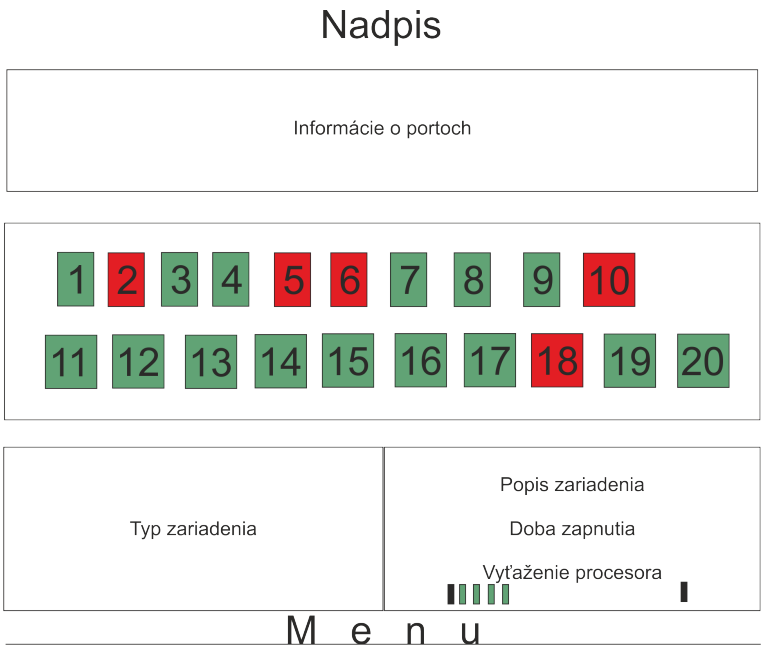
Po inicializácií spojenia sa skontroluje, či všetko prebehlo v poriadku a program sa rozhoduje, či nenastala chyba a ak áno, ukončí program. Rozhodnutie je znázornené kosoštvorcom s podmienkami po bokoch a jednou z možností bol koniec vykonávania programu, znázornený ako čierny kruh obalený kružnicou.

Po zobrazení zariadenia (prepínača) sa spustia naraz dve funkcie, každá v samostatnom vlákne. V tomto prípade bola použitá hrubá vodorovná čiara, ktorá ukazuje na obe funkcie naraz a znázorňuje rozvetvenie spracovania aplikácie.

4.5 Návrh grafického prostredia

Na sekvenčnom a aktivity diagrame je vidieť, akým spôsobom je vykonávaný beh programu, alebo načítavanie SNMP hodnôt. Avšak, aby som vedel dáta zobraziť a program interaktívne ovládať, je dôležité mať prehľadné grafické prostredie, kde každý ovládací prvok má svoje miesto a nič nieje na seba nahusto natlačené.

Preto nasledujúci Obrázok 6 zobrazuje približný vzhľad mojej aplikácie, ktorá beží len v jednom okne s dynamicky meneným obsahom.



Obr. 6: Prototyp užívateľského rozhrania

Na Obrázku 6 je rozdelená obrazovka do štyroch častí (podokien). Vo vrchnom okne sa budú nachádzať textové informácie o portoch a hneď pod ním bude grafické vyobrazenie umiestnenia portov a ich stavy. V spodnej tretine obrazovky sa budu zobrazovať informácie týkajúce sa samotného zariadenia, ako typ, popis, doba behu a vyťaženie procesora na ociachovanej stupnici. V najspodnejšej časti bude panel menu pre spúšťanie funkcií.

5 Implementácia

V tejto kapitole sa zameriam na technické detaily pri implementácií ukážkového SNMP programu, ako aj na postup nastavenia SNMP servera na požadovanom zariadení. Potom budem hovoriť o prototype architektúry vyvíjanej aplikácie. Vysvetlím a ukážem najdôležitejšie funkcie a prácu s knižnicami. Následne budem rozoberať použité implementačné nástroje, ako jazyk a vývojové prostredie.

5.1 Konfigurácia SNMP servera

Na samom začiatku je nutné povoliť SNMP agenta na zariadení. Ako popisuje kapitola 2.2.3, aby bolo možné zisťovať SNMP hodnoty z môjho testovacieho prepínača, musel som povoliť SNMP server a nastaviť autentifikačné parametre.

Vzhľadom na vysvetlené verzie protokolov v kapitole 2.5, si vyvíjaná aplikácia nevyžadovala štandard SNMPv3, tak mi stačilo nastaviť len komunitný reťazec a metódu prístupu. Uvedené príkazy sú akceptovateľné v systéme iOS, na IP zariadeniach od firmy CISCO.

```
Switch(config)#snmp-server community nazovKomunity RO
```

RO znamená prístup len na čítanie (angl. read only). Samozrejmé je dôležite, aby uvedené zariadenie malo správne nastavenú (dostupnú) IP adresu na manažment VLANe a neblokovalo SNMP porty 161 a 162.

5.2 Implementačný jazyk a použité nástroje

Na implementáciu som zvolil jazyk C, ktorý patrí medzi kompilované jazyky a dá sa obohatiť o množstvo prídavných funkcií. Kompilácia prebieha prekladačom (v systéme Linux je to Gcc), ktorý zo zdrojového kódu napísaného v jazyku C, vygeneruje spustiteľný súbor.

Ako výhody tohto jazyka uvádzam:

- vysoká pamäťová efektivita programov a výpočtová rýchlosť
- možnosť nízkoúrovňových operácií pre priamu komunikáciu s hardvérom.
- široká dostupnosť kvalitných prekladačov a knižníc.

Ďalej uvádzam, aké nástroje boli potrebné pri vývoji tohto softvéru

- Linux Ubuntu 12.10 32bit
- Code::Blocks 12.1
- GNOME Terminal 3.4.1.1
- Kompilátor gcc
- Make
- Logovací systém Syslog

- Knižnica NCURSES 5.9
- Knižnica Net-SNMP 5.7.2
- Knižnica PThread

Dokumentáciu môjho programu a teóriu som písal v typografickom systéme LaTeX. Jednotlivé diagramy som kreslil za pomoci nástroja Violet UML Editor, ktorý je šíriteľný pod voľnou GPL licenciou.

Tak, ako uvádza zdroj [12], použitie logovacieho systému **Syslog** je dôležité pre analýzu chybovosti rôznych programov fungujúcich pod Linuxom. Po pridaní hlavičky `#include <syslog.h>` do zdrojového kódu je možné využiť zápisnú funkciu `syslog()`, ktorej vstupné argumenty sa ukladajú priamo do systémového logu *syslog* uloženého v adresári /var/log/. Pred prvým zápisom je potrebne log otvoriť a určiť formát zápisu. O to sa stará funkcia `openlog()`.

Spomenuté logovanie som využil v implementovaných funkciách SNMP pri načítavaní hodnôt zo zariadenia. Pretože stav komunikácie medzi agentom a manažérom nemožno predvídať, a preto je to najpravdepodobnejšie miesto výskytu chyby. V takomto prípade sa vzniknuté chyby nevypíšu na obrazovku používateľa, ale uložia sa do systémového logu, kde si to bude môcť skúsenejší užívateľ pozrieť.

5.3 Použité prídavné knižnice

Na účel mojej aplikácie mi nestačil štandardný balík funkcií jazyka C, ale bol som nútení siahnuť po špecifickejších balíkoch, zaoberajúcich sa požadovanou tématikou, nazývané knižnice (angl. libraries), ktoré mali v sebe zabalené potrebné funkcie.

1. NCURSES
2. Net-SNMP
3. Pthread

Aby kompilátor vedel, že má možnosť použitia väčšieho rozsahu funkcií v knižniciach, tak mu je ich potrebné nalinkovať a pridať ich hlavičky do zdrojového kódu. Knižnice linkujeme s parametrom -l a hneď za tým názov knižnice. Tento link sa zadáva, ako posledný parameter prekladačovi, čiže napríklad prekladaču Gcc, ktorý kompiluje súbor main, dávame k dispozícií knižnicu netsnmp bude vyzerať takto: `gcc main.c -o main -lnetsnmp`

5.3.1 Knižnica NCURSES

Názov knižnice Ncurses je zložený zo slov new curses. Ide o voľnú softvérovú emuláciu curses System V Release 4.0. Používa terminálový formát, podporuje odsadenia, farby, zvýrazňovanie, formuláre a mapovanie funkčných kláves. Obsahuje radu vylepšení oproti BSD curses. Ncurses kód bol vyvinutý v rámci GNU / Linux. Používala sa nejakú dobu s OpenBSD ako systémová Curses knižnica na FreeBSD a NetBSD ako externý balík. Mala by byť kompatibilná s akoukoľvek distribúciou UNIXu typu ANSI / POSIX a tiež dokonca s OS / 2 Warp.

Pri implementácií projektu mali tieto funkcie najväčší výskyt a najväčšie opodstatnenie:

- `initscr(), endwin()` - štart/stop grafického módu Ncurses.
- `newwin()` - vracia ukazovateľ na obrazové okno.
- `wrefresh()` - vykresľuje okno na obrazovku.
- `printw(), mvprintw(), mvwprintw(), mvaddstr()` - výsadba textu na obrazovku, podľa zadaných parametrov.
- `attron(), attroff()` - riadenie formátovania znakov.

Zdroj: [9]

5.3.2 Knižnica Net-Snmp

Knižnica poskytuje rôzne nástroje, vzťahujúce sa k protokolu SNMP, ktoré zahŕňajú:

1. rozšíriteľného agenta.
2. nástroje na získanie alebo nastavenie hodnôt z agentov.
3. nástroje na vytváranie a spracovanie SNMP Trap udalostí.
4. grafický prehliadač MIB založený na Perl/TK.

Tento balíček je pôvodne vyvinutý na Carnegie Mellon univerzite, kde bolo obohatený SNMP integráciou (verzia 2.1.2.1) a odvtedy sa najvýznamnejšie rozvíjal. Je dostupná pre mnoho UNIXových variácií a taktiež pre Microsoft Windows.

Pri programovaní môjho SNMP programu boli z tejto knižnice najčastejšie použité tieto funkcie:

- `snmp_synch_response()` - synchrónne posiela SNMP správy.
- `snmp_pdu_create(), snmp_free_pdu()` - vytvára a uvoľňuje v pamäti SNMP správy.
- `snmp_add_null_var()` - vypĺňa správy prázdnymi údajmi.
- `snmp_parse_oid()` - integruje textový identifikátor požadovanej hodnoty do objektu OID.

Zdroj: [10]

5.3.3 Knižnica Pthread

Táto POSIX knižnica je založená na API štandarde vlákien pre jazyk C/C++. To umožňuje, aby jeden proces vytváral súbežné nové procesné toky. Najviac je to efektívne s viacerými procesormi alebo s viacjadrovými systémami, kde prevádzka procesného toku môže byť rozvrhnutá na inom procesore a tým získava rýchlosť cez paralelný beh alebo distribuované spracovanie.

Vlákna vyžadujú menšie nároky na vytvorenie nového procesu, ako vetvenia, pretože systém neinicializuje novú virtuálnu pamäť a prostredie. Všetky vlákna v rámci procesu zdieľajú rovnaký adresový priestor. Vláknu treba definovať funkciu a jej argumenty, ktoré sa budú v nej spracovávať.

Zdroj: [11]

Účelom použitia vlákien POSIX v programe je spustiť softvér rýchlejšie, a preto som túto vlastnosť využil pri štarte programu na rýchlejšie načítanie SNMP hodnôt zo zariadenia. Za pomoci vlákien som simultánne spustil dve plniace funkcie, ako je vidno z aktivity diagramu na Obrázku 5.

Pri programovaní môjho SNMP programu boli z tejto knižnice najčastejšie použité tieto funkcie:

- `pthread_create(), pthread_cancel()` - spúšťa/ukončuje zadefinované vlákno s parametrom.
- `pthread_join()` - čaká na dokončenie vykonávania zadefinovaného vlákna.
- `pthread_mutex_lock()` - pozastavenie vykonávania zadefinovaného vlákna.

Vlákno mám použité aj na nekonečný cyklus načítavania percentuálneho vyťaženia procesora. V tejto funkcii som musel použiť takzvaný semafor, kde pozastavím beh vlákna na určitú dobu. Podrobnejšie v nižšie uvedenom Výpise 1.

```
void waiting(int seconds)
{
    struct timeval now;
    struct timespec WaitingTime;

    gettimeofday(&now,NULL);
    WaitingTime.tv_sec = now.tv_sec + seconds;
    WaitingTime.tv_nsec = now.tv_usec;

    pthread_mutex_lock(&threadMutex);
    pthread_cond_timedwait(&threadCondition, &threadMutex, &WaitingTime);
    pthread_mutex_unlock(&threadMutex);
}
```

Výpis 1: Funkcia na pozastavenie behu vlákna

Vo Výpise 1 je ukázané použitie zámku `pthread_mutex_t threadMutex` a vláknovej podmienky `pthread_cond_t threadCondition`. Pomocou zámku procesor pozastaví vlákno a spustí ho až po splnení podmienky, v mojom prípade časovej. Funkciu `waiting` som si vytvoril na časový rozostup pri neustálych aktualizáciách dát.

5.4 Spôsoby implementácie niektorých funkcií

Pri pripájaní na zariadenie sa vytvára SNMP pripojenie, ktorému je nutné nastaviť autentifikačné parametre a môže sa začať využívať. Postup vytvorenia spojenia je ukázaný vo Výpise 2.

```
void sessionCreate(struct snmp_session **pp_session, char *community, char *address, int
     version)
{
    struct snmp_session session;
    init_snmp("ProgramSNMP");
    snmp_sess_init(&session);
    session.peername = address;
    session.version = version;
    session.community = community;
    session.community_len = strlen(session.community);
    *pp_session = snmp_open(&session);
}
```

Výpis 2: Funkcia na vytvorenie SNMP pripojenia

Vyššie uvedená funkcia vo Výpise 2 dostáva parametre na nastavenie autentifikačných hodnôt lokálneho SNMP pripojenia. Toto pripojenie otvorím funkciou `snmp_open()`, ktorá mi vrátila pamäťový ukazovateľ na vytvorene SNMP pripojenie. Tento ukazovateľ používam pri každej SNMP žiadosti, ako SNMP Walk a SNMP get. Toto pripojenie je nutne zavrieť pri ukončení programu metódou `snmp_close()`.

V nasledujúcich pár riadkoch kódu ukážem, akým spôsobom ukladám získané dáta do pamäte. Používam na to plniace (angl. fill) funkcie, ktoré mi získane hodnoty z SNMP požiadaviek uložia do štruktúr. Každý sieťový prepínač, alebo smerovač má rôzne vlastnosti, preto som tie najzákladnejšie zoskupil do jednej štruktúry, aby sa ňou dalo lepšie manipulovať.

```
void fillSysVariables (SysInfo_variable *sys)
{
    int i;
    sys->desc = getRequest(SYSdesc,&p_session);
    sys->type = getRequest(SYStype,&p_session);
    sys->uptime = getRequest(SYSuptime,&p_session);
    sys->ETHcount = ETHcounter;
    // cyklus na odstranenie prebytocnych riadkov
    for(i=0;i<strlen(sys->type);i++)
    {
        if(sys->type[i]=='\n')
        sys->type[i]=' ';
    }
}
```

Výpis 3: Funkcia na uloženie informácií o zariadení

Funkcia `fillSysVariables()` vo Výpise 3 dostáva cez parameter systémovú štruktúru, do ktorej vkladá získané hodnoty z funkcie `getRequest()`. Na základe zadefinovaného *OID* reťazca a SNMP pripojenia, táto funkcia vracia potrebnú hodnotu.

V niektorých prípadoch majú prepínače vlastnosť *typ* rozloženú na viaceré riadky, z toho dôvodu som zvolil cyklus na prechádzanie tejto hodnoty, aby sa prebytočný znak riadku `'\n'` nahradil medzerou.

5.5 Prototyp architektúry systému

Vyvíjaný program bol navrhnutý na návrhovom vzore MVC, kde sú jednotlivé vrstvy oddelené na logickej úrovni. V mojom prípade sú to jednotlivé funkcie zoskupené do súborov, ktoré majú spoločný cieľ, sú samostatne upravovateľné a bola dosiahnuteľná minimalizácia závislostí.

- **VIEW** - o zobrazovanie sa stará knižnica NCURSES, ktorej funkcie sú hlavne použité v súbore GUI.c, kde sú implementované v zobrazovacích funkciách ako `showSwitch()`, `startScrInit()`, `showSystem()`, `showAbout()` a `showInterface()`.
- **MODEL** - o získavanie dát zo zariadenia sa stará hlavne knižnica Net-SNMP, ktorej funkcie sú použité hlavne v súbore getOperationsSNMP.c, kde sú implementované v čítacích funkciách `SnmpWalk()` a `GetRequest()`.
- **CONTROLER** - o riadenie programu sa stará hlavná časť programu main.c ktorá volá funkcie na manipuláciu s objektmi, uložených v súboroch fillStructures.c a sessionInit.c.

Vzhľadom na použitý vzor MVC boli dodržané nasledovné závislosti.

View má dosah na Model a Controler.

Model nemá na nič dosah.

Controller má dosah na Model a View.

5.6 Kompilácia a spustenie

Najbežnejší spôsob nasadenia SNMP programu je jeho nasadenie na pracovnú stanicu a následná kompilácia priamo na počítači. Taktiež je možné použiť predkompilovanú verziu, ktorú je možné získať z elektronickej verzie tohto dokumentu.

Pri samotnej kompilácií a spustení je dôležité mať nainštalované v systéme všetky knižnice spomenuté v kapitole 5.3 a to konkrétne Net-SNMP a NCURSES, zvyšné sú už obsiahnuté v základnej inštalácií Linuxu.

Knižnica NET-SNMP má pre všetky podporované platformy vývojarské balíky dostupné na webových stránkach projektu [15]. Postup akým je možné nainštalovať tento balík je zverejnený na internetovej adrese [13]. V mojom prípade som postupoval podľa návodu určenom pre Ubuntu, dostupného na web stránke [14].[1]

Knižnicu NCURSES je možné v Linuxovej distribúcií Debian a jeho odvodených klonoch (napr. Ubuntu), jednoducho nainštalovať, nakoľko sa nachádza v ich repozitároch, pod super užívateľom zadaním príkazu

```
sudo apt-get install libncurses5-dev
```

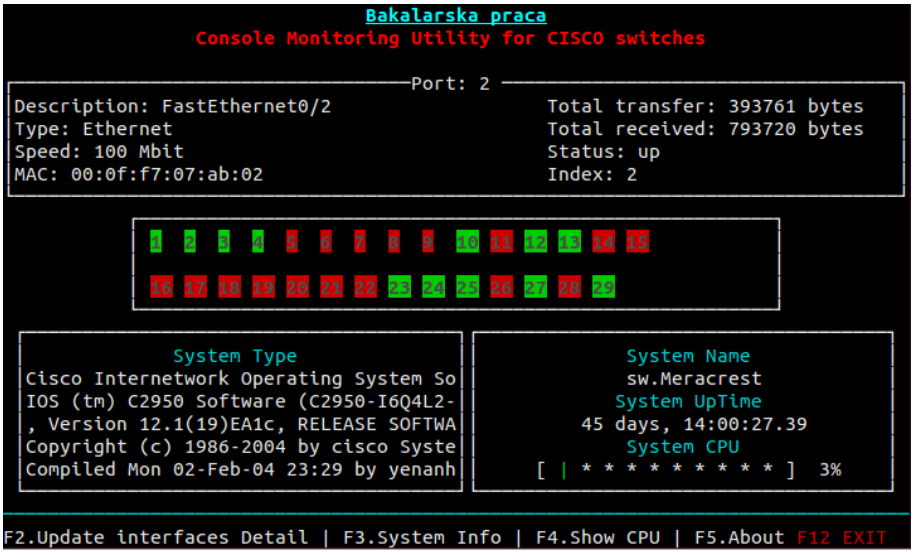
[1] Uvedené webové referencie sa nachádzajú zozname použitej literatúry na konci tejto práce.

Následne sa všetko rozbalí, inštalácia prebieha automaticky a nieje potrebná ďalšia konfigurácia. Po nainštalovaní oboch knižníc je možné spustiť kompiláciu, ako je uvedené nižšie.

```
make
./main -v<číslo verzie> -c<názov komunity> <ip adresa>
```

Výpis 4: Spustenie SNMP programu s kompiláciou

Spustí sa prvá a jediná obrazovka, tak ako je vidieť na Obrázku 7. V jeho spodnej časti je navigačné menu, popisujúce funkcionalitu jednotlivých funkčných kláves a v strede je grafický prototyp zariadenia znázorňujúce počet a stav portov. Zelená farba indikuje, že je daný port pripojený a červená, že je odpojený. Dosiahol som zobrazenie všetkých grafických prvkov tak, ako som predbežne navrhol na Obrázku 6 a popísal v kapitole 4.5, kde som opisoval návrh grafického prostredia.



Obr. 7: Obrazovka SNMP programu

5.7 Perspektíva ďalšieho vývoja

V programovaní mojej monitorovacej utility je možné pokračovať rozšírením oboru hodnôt získavaných informácií, nakoľko MIB databáza poskytuje široké možnosti získavania dát.

Ďalším prínosom by bolo vytvorenie samostatného vlákna pre proces neustáleho čítania tečúcich paketov cez práve zobrazovaný port, ktoré by navýšilo výkon aplikácie. Pri tomto riešení by bolo vhodné premenné zobrazovaných čísiel oddeliť od poľa štruktúr portov do samostatnej číselnej hodnoty. Táto hodnota by bola získavaná v nekonečnom cykle pomocou funkcie `getRequest()`, vyhli by sme sa priamemu prístupu k štruktúram portov a nemuseli by sme tak implementovať semafor. Totižto, mohlo by dojsť k rovnakému čítaniu zo štruktúry, ktorá by bola práve spracovávaná hlavným vláknom.

Ďalšou prácou by mohla byť modifikácia SNMP prístupu na prípojnom zariadení. Zmenilo by sa povolenie čítania na povolenie zápisu. V kapitole 5.1 sme sa venovali konfigurácií SNMP servera a pri zadávaní autentifikačného príkazu sa volí metóda prístupu. Zmenilo by sa **RO** (angl. Read-only) na **RW** (angl. Read-write). Umožnilo by nám to implementovať Set funkcie SNMP protokolu a pomocou neho nastavovať hodnoty na diaľku.

V grafickom rozhraní by bolo vhodné implementovať schopnosť za behu programu dynamicky meniť obsah okna v závislosti na jeho veľkosti. V konzolách bežiacich pod oknami bude možné zväčšiť vykresľovaný priestor z väčšou efektivitou. Knižnicu NCurses, ktorá sa stará o zobrazovanie, je možné rozšíriť o knižnicu Menu.h a knižnicu Form.h. Funkcie z balíka Menu.h by sa starali o precízne ovládanie aplikácie a funkcie z balíka Form.h by obsahovali programové vybavenie na formuláre slúžiace na zadávanie hodnôt (najme pri nastaviteľných operáciach).

6 Záver

V tejto práci som prešiel technológie, ktoré uľahčujú prácu sieťovým administrátorom. Každý nástroj, ktorý vykonáva za nás úlohy predstavuje urýchlenie práce, lebo vykonáva svoje činnosti automatizovane. V mojom prípade išlo o načítavanie údajov zo sieťového zariadenia a ich zobrazenia na obrazovke počítačov. Tieto zariadenia používali na vzdialené ovládanie sieťový riadiaci protokol SNMP, pomocou ktorého komunikovali v roly SNMP agenta s mojím programom v roly SNMP manažéra. Na základe MIB databázy bolo možné vytiahnuť zo zariadenia takmer akýkoľvek udaj, na ktorý nám výrobca zariadenia uverejnil a implementoval identifikátor.

Dôležitou súčasťou práce je spôsob používania rozširujúcich balíčkov programovacieho jazyka. Tieto balíčky nazvané knižnice, mi pomohli k vývoju programu do podoby, ako vyzerá teraz. Každá knižnica obohatila program o jednu dôležitú časť. Spôsob nasadenia týchto knižníc ma priviedlo k myšlienke, že takto programátor dostáva obrovské možnosti vývoja a použitý programovací jazyk C sa tak stáva vysoko flexibilným nástrojom. Pri programovaní SNMP utility som postupne získaval informácie o týchto nástrojoch, skúšal ich použitie na jednoduchších príkladoch, a potom postupne implementoval do môjho projektu. Prešiel som tak od stavebných základov programovacieho jazyka C, cez prácu s ukazovateľmi, až po jeho využitie vlákien. Vďaka týmto krokom a preštudovaní dostupnej literatúry bolo možné dostať program do takej podoby, v akej je dnes.

Vďaka spomenutým technologickým prostriedkom bolo možné vyvinúť program, ktorý je predstavený v kapitolách. 4, 5 a v ich podkapitolách. Metodika spoločne s implementáciou nástroja na získavanie údajov zo sieťového zariadenia predstavuje využitie dostupných technológií vedúcich k lepšej a efektívnejšej práce z prvkami IT siete. Vzhľadom na to, bude implementovaná SNMP utilita využitá vo firme, v ktorej pracujem. Keďže sme internetový poskytovateľ, tak prichádzame do každodenného styku s prepínačmi od spoločnosti CISCO a použitie tejto aplikácie nám ušetrí čas, ktorý je pre zamestnávateľov a ich klientov najvzácnejší.

Ľubomír Wenzl

7 Literatúra

[1] MAURO, Douglas, and SCHMIDT, Kevin James. *Essential Snmp.* First ed. USA: O'Reilly and Associates, 2001. 14-16 s. ISBN 0-596-00020-0.

[2] ZOHO Corporation, *What is SNMP — SNMP tutorial — SNMP basics :: OpManager Tutorials [online]*, last revision 2013 [vid. 28.02.2013]. Dostupné z: <http://www.manageengine.com/network-monitoring/what-is-snmp.html>

[3] SHIMONSKY, Robert, *Introduction to the Simple Network Management Protocol (SNMP) Part 1 [online]*, last revision 28.11.2005 [vid. 28.02.2013]. Dostupné z: <http://www.windowsnetworking.com/articles-tutorials/network-protocols/Introduction-SNMP-Part1.html>

[4] ODOM, Wendell; HEALY, Rus and DONOHUE, Denise. *CISCO CCIE Routing and Switching.* Fourth ed. USA: Cisco Press, 2010. 155-185 s. ISBN 1-58705-980-0.

[5] PAESSLER AG, *PRTG - network monitoring using SNMP, NetFlow, WMI and more [online]*, last revision 2013 [vid. 28.02.2013]. Dostupné z: <http://www.paessler.com/prtg/features>

[6] CACTI Group, *Cacti® - The Complete RRDTool-based Graphing Solution [online]*, last revision 2012 [vid. 28.02.2013]. Dostupné z: <http://www.cacti.net/what_is_cacti.php>

[7] NAGIOS Enterprises, *Nagios - Nagios Overview [online]*, last revision 2013 [vid. 28.02.2013]. Dostupné z: <http://www.nagios.org/about/overview/>

[8] HW group, *HWg případové studie [online]*, last revision 2013 [vid. 28.02.2013]. Dostupné z: <http://www.hw-group.com/case_studies/index_cz.html>

[9] DICKEY, Thomas *Announcing ncurses 5.9 - GNU Project - Free Software Foundation (FSF) [online]*, last revision 4.4.2011 [vid. 28.02.2013]. Dostupné z: <http://www.gnu.org/software/ncurses/ncurses.html>

[10] SOURCEFORGE, *FAQ:General 01 - Net-SNMP Wiki [online]*, last revision 28.12.2006 [vid. 28.02.2013]. Dostupné z: <http://www.net-snmp.org/wiki/index.php/FAQ:General_01>

[11] IPPOLITO Greg, *Linux Tutorial: POSIX Threads [online]*, last revision 2012 [vid. 28.02.2013]. Dostupné z: <http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html>

[12] kolektiv autorů. *Linux - Dokumentační projekt.* 3. vydání. Brno: Computer Press, 2001. 152 s. ISBN 80-7226-761-2.

[13] SOURCEFORGE, *Net-SNMP [online]*, last revision 26.05.2011 [vid. 28.02.2013]. Dostupné z: <http://www.net-snmp.org/docs/INSTALL.html>

[14] SOURCEFORGE, *Net-Snmp on Ubuntu - Net-SNMP Wiki [online]*, last revision 16.02.2011 [vid. 28.02.2013]. Dostupné z: <http://www.net-snmp.org/wiki/index.php/Net-Snmp_on_Ubuntu>

[15] SOURCEFORGE, *Net-SNMP [online]*, last revision 24.10.2012 [vid. 28.02.2013]. Dostupné z: <http://www.net-snmp.org/download.html>

[16] BYTESPHERE, *Free Cisco MIB Download Search MIBs Download MIBs for Cisco Network Management - OiDViEW [online]*, last revision 01.12.2012 [vid. 28.02.2013]. Dostupné z: <http://www.oidview.com/mibs/9/md-9-1.html>

[17] SOLARWINDS®, *ipMonitor Support Portal :: SNMP Center [online]*, last revision 2012 [vid. 28.02.2013]. Dostupné z: <https://support.ipmonitor.com/snmp_center.aspx>

[18] CISCO Systems, *Cisco SNMP Object Navigator [online]*, last revision 2013 [vid. 28.02.2013]. Dostupné z: <http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en>

A Elektronická príloha

Disk DVD obsahuje okrem PDF verzie tohto dokumentu aj zdrojové kódy ukážkovej SNMP aplikácie.